VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

3D rekonstrukce okolí vozidla

3D Reconstruction of the Vehicle Neighborhood

2014 Tomáš Worek

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Tomáš Worek**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: 3D rekonstrukce okolí vozidla
3D Reconstruction of the Vehicle Neighbourhood

Zásady pro vypracování:

Cílem diplomové práce je navrhnout postup a naimplementovat software pro 3D rekonstrukci okolí jedoucího vozidla.

V diplomové práci proveďte následující:
1. Seznamte se s technikami, kterých lze pro řešení problému použít.
2. Navrhněte a realizujte postup pro získávání vhodných sekvencí z jedoucího vozidla.
3. Navrhněte a realizujte metodu pro získávání zájmových bodů a jejich sledování v jednotlivých snímcích.
4. Navrhněte a realizujte metodu pro výpočet fundamentální matice mezi vybranými snímky.
5. Navrhněte a realizujte metodu pro 3D rekonstrukci vybraných bodů.
6. Funkčnost navržených a realizovaných postupů řádně prověřte a výsledky zdokumentujte.

Programátorské práce proveďte v C/C++; můžete také využívat dostupných knihoven dle vaší volby (např. knihovny OpenCV).

Seznam doporučené odborné literatury:

Podle pokynů vedoucího diplomové práce.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **doc. Dr. Ing. Eduard Sojka**

Datum zadání: 01.09.2013

Datum odevzdání: 07.05.2014

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.*

V Ostravě 7. května 2014 .....

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 7. května 2014 .....

Rád bych na tomto místě poděkoval všem, kteří mi s prací pomohli, protože bez nich by tato práce nevznikla.

Abstrakt

V této práci je popsán systém pro 3D rekonstrukci okolí vozidla. Jednotlivé části 3D rekonstrukce, jako detekce a sledování zájmových bodů nebo jejich triangulace, jsou navrhnuty, implementovány a popsány jejich úskalí. Pro detekci bodů jsou použity detektory FAST a GFTT. Sledování bodů mezi snímky probíhá výpočtem optického toku pomocí pyramidové metody Lucas-Kanade. Ke triangulaci bodů je použit algoritmus lineární triangulace nejmenších čtverců. Výsledná vizualizace probíhá formou hloubkové mapy a interaktivního vizualizéru.

Klíčová slova: 3D, rekonstrukce, vozidlo, monokulární

Abstract

This work describes a system for 3D Reconstruction of the Vehicle Neighborhood. Each part of 3D reconstruction, such as detection and tracking features or triangulation is designed and implemented, and are described their issues. For the detection of the features are used FAST and GFTT detectors. Tracking points between the images is carried out by calculating the optical flow using pyramidal Lucas-Kanade algorithm. For triangulation is used the linear least-square method. Final visualization has a form depth map and interactive visualizer.

Keywords: 3D, reconstruction, vehicle, monocular

Seznam použitých zkratek a symbolů

OpenCV	–	Open source Computer Vision
PTAM	–	Parallel tracking and mapping
SLAM	–	Simultaneous localization and mapping
PCL	–	Point Cloud Library
GFTT	–	Good Features To Track
PCL	–	RANdom SAmple Consensus

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

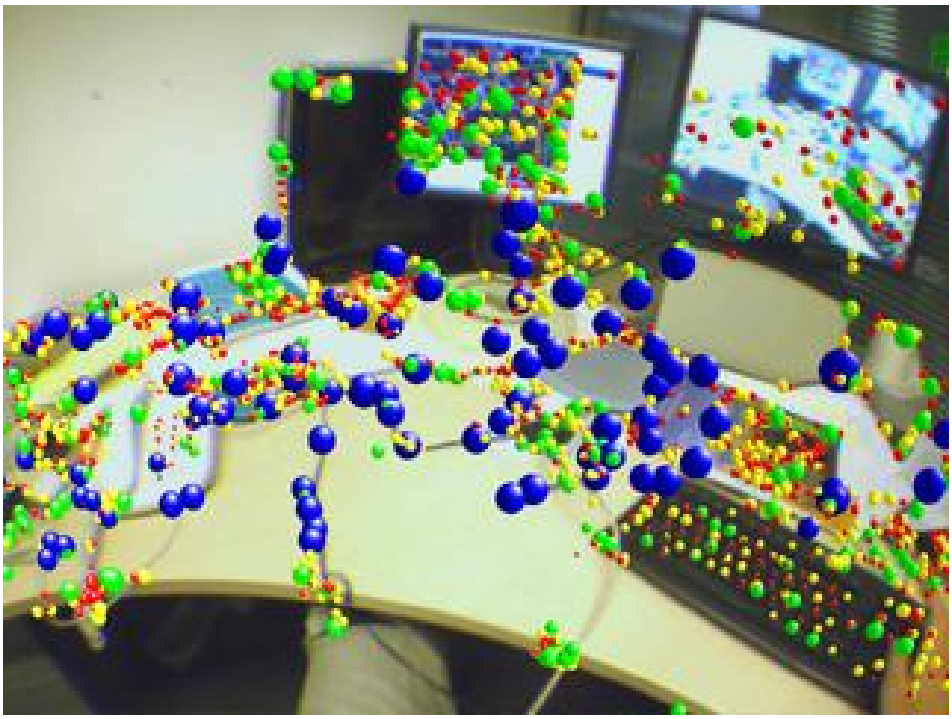
1 Úvod

3D rekonstrukce okolí vozidla má v dnešní době mnoho potenciálních využití, od parkovacích systémů až po trojrozměrnou alternativu Street View.

Cílem této práce je navrhnout a implementovat jednotlivé části 3D rekonstrukce okolí vozidla jako např. detekce bodů, jejich sledování a rekonstrukci těchto bodů. Implementaci těchto částí poté popsat a upozornit na případná úskalí a problémy s nimi spojené.

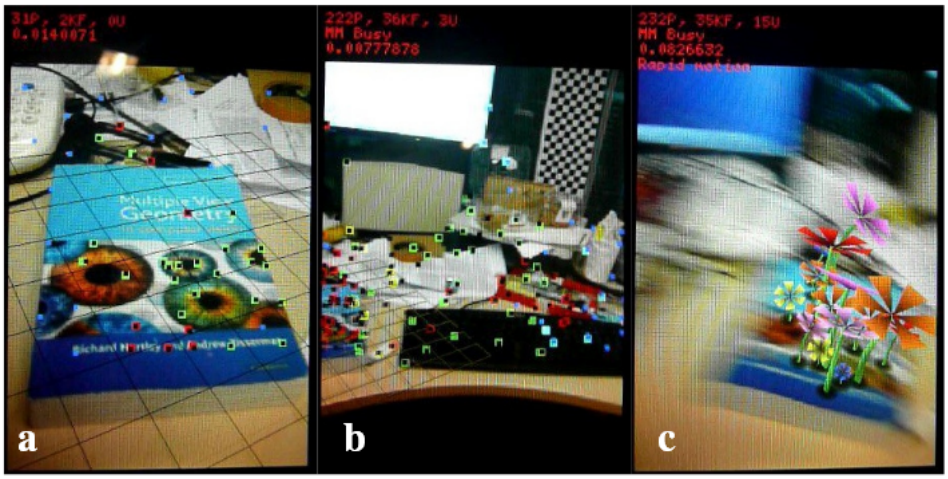
2 Inspirační a podkladové práce

V roce 2007 prezentovali svou práci, viz [1], pánové Klein a Murray na téma paralelního sledování a mapování (PTAM) malých prostor pro rozšířenou realitu. V této práci se soustředili na porovnání se systémem simultálního lokalizování a mapování (SLAM). Snímek z aplikace této práce lze vidět na obrázku 1.



Obrázek 1: Ukázka paralelního sledování a mapování malého prostoru

Na tuto práci navázali v roce 2009 v práci [2] vylepšeným systémem na sledování a mapování pomocí mobilní kamery. Snímek z této práce je pak vidět na obrázku 2.



Obrázek 2: Ukázka paralelního sledování a mapování malého prostoru

3 Teoretický podklad

3.1 Systém snímání

Existují různé systémy pro snímání sekvence snímků určených k následné rekonstrukci. Počínaje monokulárním systémem, který je založen na snímání z jedné kamery, po stereoskopický systém, kterému odpovídá paralelní snímání ze dvou navzájem posunutých kamer, až po multiskopické snímací systémy, skládající se ze soustavy více kamer rozmístěných typicky na pevné pohyblivé mřížce nebo na obvodovém kruhu umístěném okolo snímaného objektu.

3.1.1 Monokulární systém

Monokulární systém snímání je tvořen pouze jedinou kamerou. Jelikož je však pro 3D rekonstrukci obrazu nutné mít alespoň dva snímky okolí, je potřeba tohoto docílit pohybem kamery. Na dvou snímcích o jiné pořizovací pozici je pak možné založit 3D rekonstrukci.

Výhody a nevýhody monokulárního systém jsou opačné než u stereoskopického systému.

Monokulární systém nepotřebuje žádnou dodatečnou konstrukci pro připojení další kamery. Není tak potřeba pořizovat zvláštní vybavení pro tento systém snímání, využijeme totiž to, co už většinou vlastníme. At'už se jedná o fotoaparát s možností nahrávání videosekvence, nebo v dnešní době tolik rozšířené chytré telefony.

Nevýhodou však je, že je poměrně obtížnější rekonstruovat obraz z monokulárního systému, protože se nemůžeme spolehnout na stejnou vzájemnou prostorovou pozici všech snímaných objektů. Jelikož jsou dva klíčové snímky, na kterých zakládáme 3D rekonstrukci, pořízené v jiném čase, mohlo dojít ke změně pozice některého z nich vzhledem k ostatním. V případě pořizování snímků z jedoucího vozidla patří mezi objekty, které změnily svou pozici např. jiné pohybující se vozidlo či chodec.

3.1.2 Stereoskopický systém

Stereoskopický snímací systém je nám - lidem - nejblíže. Stejně jako člověk používá k rozpoznání vzdálenosti objektu dvě oči, tak i stereoskopický systém je založen na soustavě dvou souběžně snímajících kamer umístěných paralelně vedle sebe. Tyto kamery

mají zpravidla pevnou vzájemnou pozici, což je zajištěno buďto přímo z výroby, nebo dodatečným spojením dvou kamer či snímačů pomocí pevné konstrukce.

Tento systém má jednoznačnou výhodu v tom, že k rekonstrukci je potřeba pouze pořídit snímek ve stejnou dobu u obou spojených kamer. Tak vzniknou 2 snímky pořízené ve stejném čase s mírně rozdílným místem pořízení. Při provedení kalibrace pozic těchto dvou kamer je už pak relativně jednoduché rekonstruovat okolí kamer.

Nevýhodou tohoto systému je však ten fakt, že k 3D rekonstrukci je potřeba dvou kamer spojených do jedné pevné konstrukce, viz obrázek 3.

Obrázek 3: Ukázka stereoskopického systému

3.2 Detekce zájmových bodů

V obraze je nutno detekovat body, které odpovídají bodu v prostoru promítnutému do 2D obrazu kamery, a které následně budeme sledovat napříč videosekvencí. Takové body se nazývají zájmové.

Zájmové body by měly být takové body v obraze, které jsou snadno identifikovatelné, vyskytují se v obraze po celou dobu sledování a jsou jednoznačné.

Pro detekci bodů jsem ve svém řešení použil GFTT detektor a FAST detektor.

3.2.1 GFTT detektor

GFTT detektor pracuje v následujících krocích:

1. Spočítá míru kvality rohů pro všechny pixely v obraze pomocí Harrisova detektoru hran.
2. Potlačí okolí lokálních maxim kvůli zamezení detekce příliš blízkých bodů.
3. Odstraní rohy, které nemají dostatečně velkou hodnotu R, viz kapitola 3.2.1.1.
4. Zbývající rohy jsou seřazeny sestupně dle kvality.
5. Odstraní rohy, které mají v zadané blízkosti roh s lepší kvalitou.

3.2.1.1 Harrisův detektor

Harrisův detektor prohledává okénko $w(x, y)$ s posunem u ve směru x a posunem v ve směru y a vyhledává změnu gradientu. Tuto funkci můžeme zapsat jako

$$E\left(u, v\right) = \sum_{x,y} w\left(x, y\right) \left[I\left(x + u, y + v\right) - I(x, y)\right]^2 \tag{1}$$

kde

- $w\left(x, y\right)$ je okénko na pozici (x, y),
- $I\left(x, y\right)$ je intenzita na pozici (x, y),
- $I\left(x + u, y + v\right)$ je intenzita pohybujícího se okna na pozici $(x + u, y + v)$.

Hledáme-li tedy okénka s rohy, hledáme vlastně okénka s velkou změnou intenzity. Proto je potřeba maximalizovat předešlou rovnici, konkrétně:

$$\sum_{x,y} \left[I\left(x + u, y + v\right) - I(x, y)\right]^2 \tag{2}$$

pomocí Taylorova rozvoje pak máme

$$E\left(u, v\right) \approx \sum_{x,y} \left[I\left(x, y\right) + uI_x + vI_y - I(x, y)\right]^2 \tag{3}$$

po odečtení a umocnění

$$E(u,v) \approx \sum_{x,y} u^2 I_x^2 + 2uvI_xI_y + v^2 I_y^2 \tag{4}$$

což můžeme zapsat v maticovém formátu

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \left(\sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix} \tag{5}$$

Označíme-li pak

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \tag{6}$$

dostaneme rovnici

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \tag{7}$$

Pro každé okénko je pak počítána hodnota R jako

$$R = \lambda_1\lambda_2 - k \cdot (\lambda_1 + \lambda_2)^2 \tag{8}$$

kde

- λ_1 a λ_2 jsou vlastní čísla matice M,
- k je práh kvality rohu.

Výsledné okénko pak považujeme za:

plochu, pokud je hodnota $|R|$ malá, tj. když vlastní čísla λ_1 a λ_2 jsou malá,

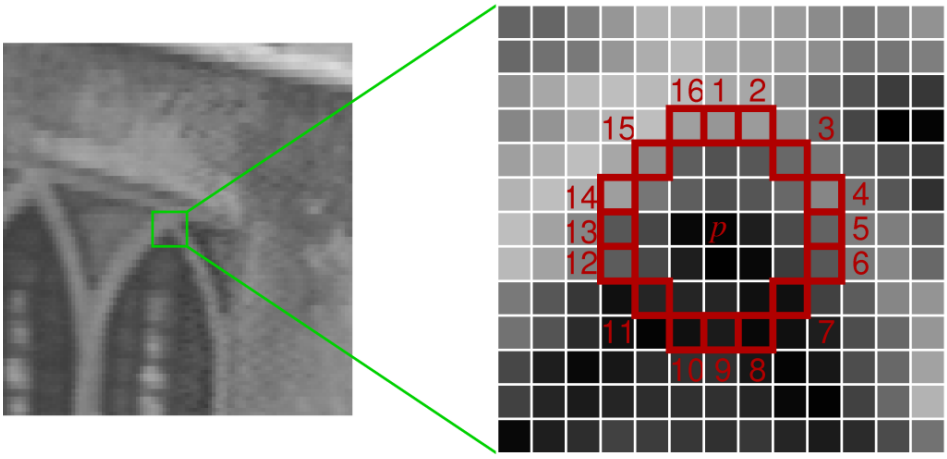
hranu, pokud je hodnota $R < 0$, tj. když $\lambda_1 >> \lambda_2$ nebo $\lambda_2 >> \lambda_1$,

roh, pokud je hodnota R velká, tj. když λ_1 a λ_2 jsou velká a $\lambda_1 \sim \lambda_2$.

3.2.2 FAST detektor

Vstupem FAST detektoru je parametr threshold, tedy jakýsi práh či mez. Detektor pak funguje tak, že pro všechny body v obraze zkoumá okolní kružnici skládající se ze 16 bodů, viz obrázek 4.

Centrální bod p je pak detekován právě tehdy, pokud existuje posloupnost alespoň 9 bodů ze 16, které jsou světlejší než bod p o více než nastavený práh detekce.



Obrázek 4: Prohledávané okolí FAST detektoru.

3.3 Sledování zájmových bodů

Ke sledování zájmových bodů jsem v řešení použil výpočet optického toku.

Optický tok určuje směr a rychlost pohybu všech nebo vybraných bodů v obrazu mezi dvěma snímky. Pro každý bod je pak výsledkem výpočtu optického toku vektor, který tento směr a rychlost pohybu udává. Tento vektor pak odpovídá sledování bodu z jednoho snímku na snímek následující.

3.3.1 Metoda Lucas-Kanade

Výpočet optického toku metodou Lucas-Kanade má dvě základní podmínky:

- Světlost bodu je v čase konstantní.
- Okolí bodu má stejný pohyb.

Rovnice konstatní světlosti je dána jako

$$I(x, y, t) = I(x + u, y + v, t + 1) \tag{9}$$

kde

- x a y je pozice sledovaného bodu,
- u a v je posun sledovaného bodu,

- $I(x, y, t)$ je světlost bodu (x, y) v čase t.

Pomocí Taylorova rozvoje $I(x+u, y+v, t+1)$ na $I(x, y, t)$ linearizujeme pravou stranu rovnice (9).

$$I(x+u, y+v, t+1) \approx I(x, y, t) + I_x \cdot u + I_y \cdot v + I_t \tag{10}$$

$$I(x+u, y+v, t+1) - I(x, y, t) \approx I_x \cdot u + I_y \cdot v + I_t \tag{11}$$

po dosazení do rovnice (9) máme

$$I_x \cdot u + I_y \cdot v + I_t \approx 0 \tag{12}$$

což můžeme zapsat maticově jako

$$\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + I_t = 0 \tag{13}$$

Jelikož máme jednu rovnici a dvě neznámé u a v, využijeme druhou podmínku, a sice, že okolní body mají stejný posun (u, v).

Při použití okénka okolo sledovaného bodu, jež bude obsahovat dalších $N-1$ bodů, máme celkem N bodů splňujících rovnici (14).

$$\nabla I(p_i) \cdot \begin{bmatrix} u \\ v \end{bmatrix} = -I_t(p_i) \tag{14}$$

Dostaneme tedy

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_N) & I_y(p_N) \end{bmatrix} \cdot \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_N) \end{bmatrix} \tag{15}$$

Pro řešení $k = \begin{bmatrix} u \\ v \end{bmatrix}$ následně využijeme metodu nejmenších čtverců $\left(A^T \cdot A\right) \cdot k = A^T \cdot b$

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \cdot \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} \tag{16}$$

Optimálním řešením je pak taková dvojice (u, v), která nám dá pravou stranu rovnice co nejblíže nulovému vektoru. Z pravé strany rovnice (16) jde vidět, že se v ní odráží rovnice (6), a tedy, že nejlépe sledovatelné budou právě ony detekované rohy v obraze.

Při pyramidovém algoritmu Lucas-Kanade metody výpočtu optického toku, se nejdříve sestaví pyramida pro oba snímky, mezi kterými probíhá sledování. Pyramida se sestává ze snímků na každé úrovni vždy o polovinu měnších v každém rozměru. Takových úrovní je typicky nejvýše 4. Např. snímek o rozměru 640x480 má na první úrovni rozměr 320x240, a na čtvrté úrovni už pouze 40x30.

Sledování pak probíhá z nějvyšší úrovně, tj. nejmenších snímků, až po snímky původního rozměru na nulté úrovni pyramidy.

3.4 Fundamentální a esenciální matice

Fundamentální matice je čtvercová matice o rozměru 3x3, která vyjadřuje vztah mezi korespondujícími body dvou snímků. Z pohledu epipolární geometrie lze na fundamentální matici nahlížet také jako na matici, díky níž k určitému bodu na jednom snímku získáme odpovídající epipolární línii na snímku druhém. Tato linie pak určuje polohu daného bodu na druhém snímku.

Fundamentální matice mezi dvěma snímky (dvěmi pozicemi kamer) je dána splněním rovnice (17).

$$pt_2^T \cdot F \cdot pt_1 = 0 \tag{17}$$

kde

- pt_1 a pt_2 označují korespondující body na dvou snímcích sekvence,
- F je výsledná fundamentální matice.

Na výpočet fundamentální matice se používají různé algoritmy jako je 7-bodový, 8-bodový, nebo RANSAC algoritmus. 8-bodový algoritmus je popsán v kapitole 4.6.

Esenciální matice je taktéž čtvercová matice o rozměru 3x3 a taktéž vyjadřuje vztah mezi korespondujícími body dvou snímků. Obsahuje však informaci o vnitřních parametrech kamery. Její výpočet je pak dán rovnicí (18).

$$E = K^T \cdot F \cdot K \tag{18}$$

kde

- E je výsledná esenciální matice,
- K je vnitřní matice kamery,
- F je dříve vypočtená fundamentální matice.

3.5 Triangulace bodů

Triangulací bodů se rozumí proces vypočtení pozice bodu v 3D prostoru daného jeho projekcí na dvou či více obrázcích.

K vyřešení tohoto problému je nutné znát transformační matice kamer.

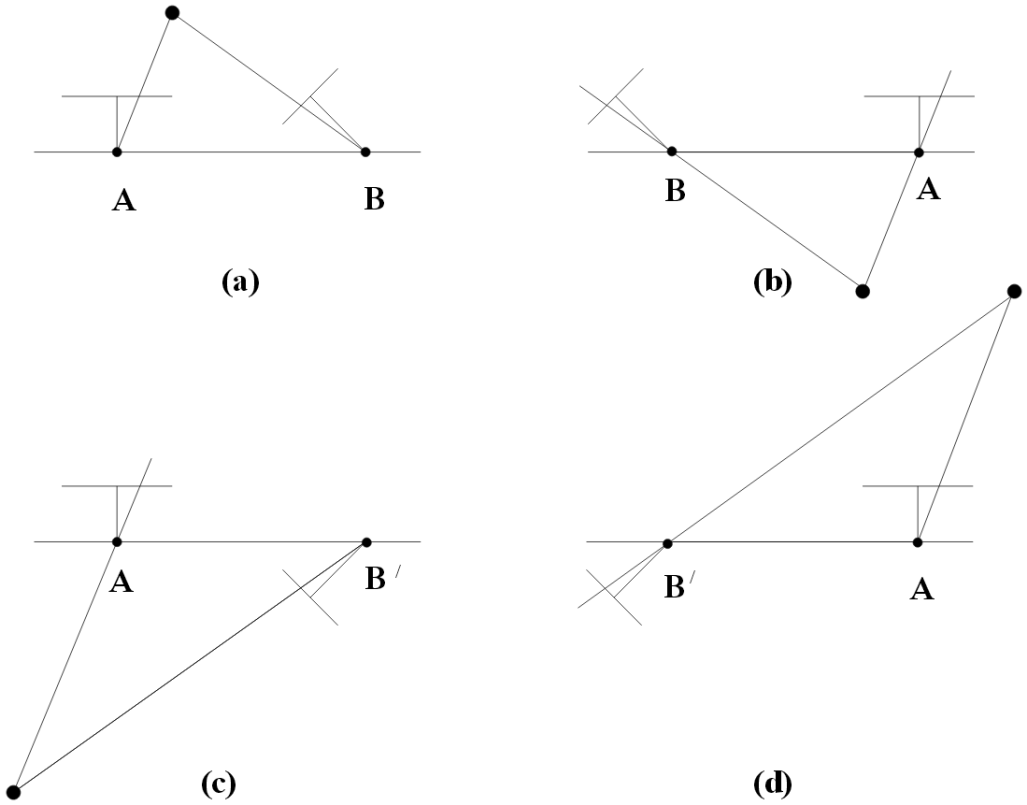
První kameře je zpravidla přisouzena defaultní pozice i rotace. Její transformační matice je tedy

$$P_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{19}$$

Transofrmační matice druhé kamery je pak dána jednou ze 4 případů, viz obrázek 5. Pravé a levé dvojice případů mají rozdílné znaménko translačního vektoru, horní a spodní dvojice se pak liší rotací druhé kamery okolo spojnice promítacích středů kamer.

Jak se vypořádat s výběrem správného řešení je popsáno v kapitole 4.7.

Z vypočtených tranformačních matic se pak nakonec provede triangulace korespondujících bodů. Ve svém řešení jsem implementoval lineární triangulaci nejmenších čtverců. Jak tato triangulace funguje popisuji v kapitole 4.8.



Obrázek 5: 4 případy vzájemné pozice kamer.

4 Vlastní řešení

Tato kapitola popisuje postup a rozhodování při implementaci vlastního řešení 3D rekonstrukce.

Postupně zde popíšu důvody výběru monokulárního kamerového systému pro rekonstrukci, dále kalibraci kamerového systému a způsob získávání videosekvence, detekování a sledování zájmových bodů a bodů určených k rekonstrukci. Následovat bude popsání výpočtu fundamentální a esenciální matice, a získání transformačních matic pomocí 4-případového rozhodování. Dále pak popíšu triangulaci rekonstruovaných bodů a zpětné určení pozice kamer.

Posledním krokem pak bude vizualizace rekonstruovaných bodů jednak formou barevné hloubkové mapy a jednak formou 3D vizualizace tzv. „point cloudu“ za pomocí vizualizéru PCL knihovny.

4.1 Monokulární systém

Monokulární systém je oproti stereoskopickému systému založen na snímání z jedné kamery. Tento systém jsem zvolil na základě dvou základních faktorů.

Jednak žijeme v době, kdy si řidiči stále více pořizují do vozidel kamery, kterými za jízdy sledují a zaznamenávají události jimiž jsou účastníky, a to ať už se jedná o kamery montované na přední sklo s výhledem dopředu, nebo montované do zadní části vozidla pro řidičův větší přehled o provozu na komunikaci za ním. Z toho důvodu má v této oblasti monokulární systém snímání oproti stereoskopickému nepoměrně větší nasazení. Pro využití mého systému tedy není potřeba nově pořizovat stereoskopický kamerový systém, ale poznatky z této práce mohou být využity při tvorbě softwaru, který se bude moci jednoduše napojit na data získaná z již pořízených kamer.

Druhým z faktorů bylo ověření a objevení nástrah při 3D rekonstruování obrazu pořízeného monokulárním systémem. Jelikož je pro 3D rekonstrukci nutné mít minimálně dva snímky odpovídajícího si okolí pořízené z různé pozice, je nutné nějak tyto dva snímky získat i z monokulárního systému. Je jasné, že využiji pohybu kamery, resp. vozidla, k získání těchto dvou snímků, bohužel z toho však plyne ten problém, že oba snímky nebudou pořízeny ve stejném čase, jako je tomu u stereoskopického systému, ale budou pořízeny v rozdílných časech. Z toho pak plyne, že se při 3D rekonstrukci nemohu spolehnout na to, že všechny objekty ve scéně nezměnily své místo mezi pořízením

prvního a druhého snímku. Takovými objekty jsou např. jiné projíždějící vozidlo nebo chodec, ale také vozidlo nesoucí kameru zachycené na kameře.

4.2 Kalibrace

Na pořízení videosekvence pro 3D rekonstrukci jsem použil fotoaparát Nikon Coolpix P330. I když má tento fotoaparát nízké sférické zkreslení, bylo i přesto žádoucí provést kalibraci obrazu.

Jednak je tedy potřeba získat koeficienty pro opravu zkresleného obrazu, ale také matici vnitřních parametrů kamery:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{20}$$

kde

- A je matice vnitřních parametrů kamery,
- (c_x, c_y) je středový bod kamery v obraze,
- f_x a f_y vyjadřují ohniskovou vzdálenost v pixelech.

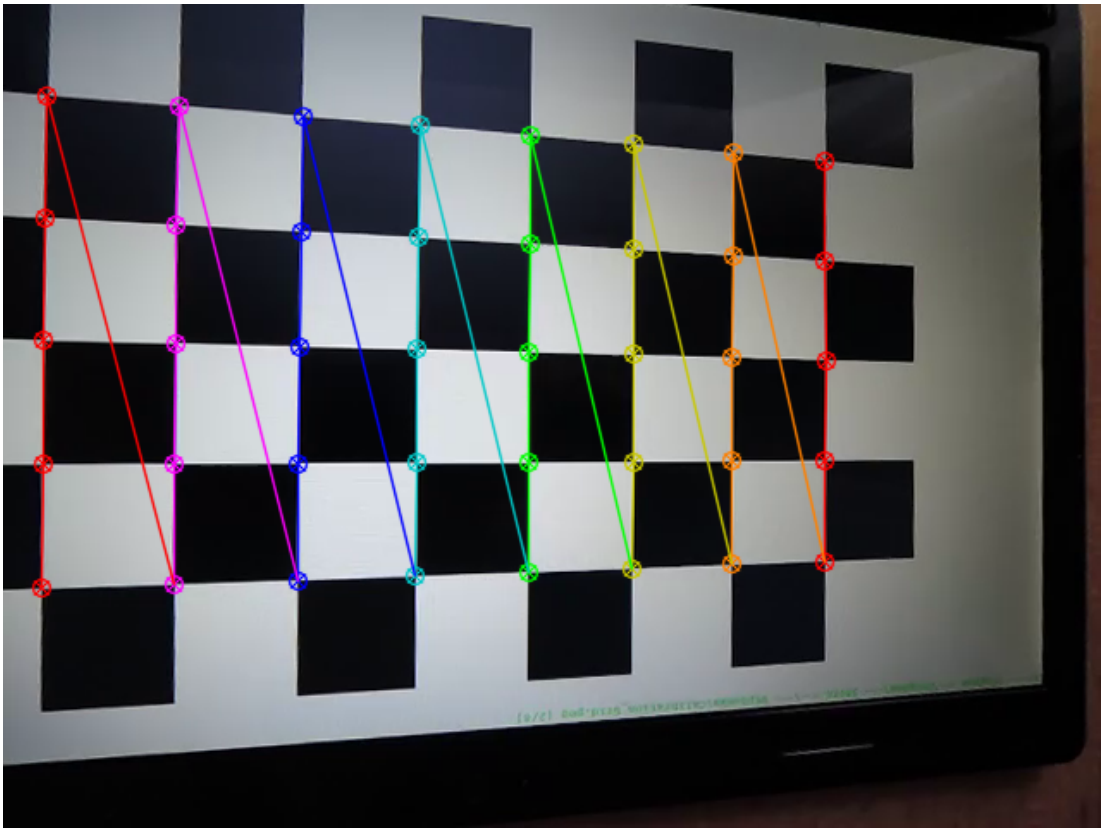
Kalibraci jsem prováděl na šachovnici 9 x 6 polí, resp. 8 x 5 spojů. Ukázka jednoho z kalibračních snímků a rozpoznaných spojů na šachovnici je vidět na obrázku 6.

4.3 Získávání videosekvence pro rekonstrukci

Při získávání videosekvence, at'už z jedoucího vozidla nebo z ruky, bylo potřeba držet se jistých omezení, abych v pozdějších krocích rekonstrukce neměl problémy s detekcí bodů a jejich sledováním napříč sekvencí, apod. Mezi tato omezení patří dostatečná ostrost obrazu a záběry bez pohybujících se objektů.

4.3.1 Ostrost obrazu

Jedním z omezení, na která bychom měli dbát je ostrý obraz. Je nutné, aby byl obraz dostatečně ostrý a měl napříč sekvencí co nejméně rozmazaných snímků, jelikož detekované



Obrázek 6: Kalibrační snímek s rozpoznanými spoji.

zájmové body mají v rozmazaných snímcích problém s určením správné pozice při jejich sledování.

Prevencí pro ostrý obraz videosekvence je jednak použití kvalitnější videokamery či fotoaparátu, v mém řešení jsem pro pořízení videosekvence použil fotoaparát Nikon Coolpix P330, a jednak natáčení videosekvence při dostatečném osvětlení. Z toho důvodu je nejlépe získat videosekvenci při jasném slunečném počasí.

4.3.2 Pohybující se objekty

Ve stereoskopickém systému snímání je možné spolehnout se na nezměněnou polohu všech objektů pořízené scény, protože snímky, ze kterých je triangulací prováděna rekonstrukce jsou pořízeny ve stejném čase.

Oproti tomu v mnou použitém monokulárním systému se na toto spolehnout nemůžu. Proto bylo nutné zamezit výskytu pohyblivých objektů ve scéně, a natáčet tak scény v místě nebo čase, kdy je takovýchto objektů čím jak nejméně.

4.3.3 Další nechtěné objekty

Dalšími zaznamenanými objekty ve scéně jsou takové, které se vzhledem ke scéně pohybují, ale vzhledem ke kameře nikoli. Takovými objekty jsou zpravidla části objektů nesoucích kamerový systém, tedy např. kapota vozidla.

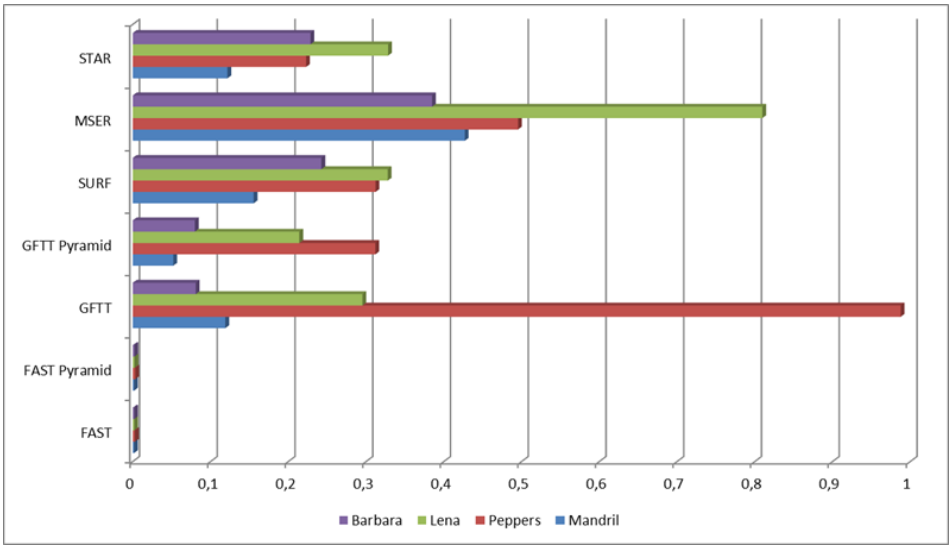
Jelikož se takové objekty nacházejí ve snímaném obraze zpravidla na okrajích jednotlivých snímků, bylo možné takové objekty vyřadit z detekce a sledování tím, že jsem před začátkem detekce v implementaci dal možnost vybrat oblast zájmu. Mimo tuto oblast pak nebude probíhat detekce zájmových bodů a bodů určených k rekonstrukci. Stejně tak budou tyto body při přechodu mimo tuto oblast při sledování napříč dalšími snímky vyřazeny z množiny rekonstruovaných bodů.

4.4 Detekování zájmových bodů

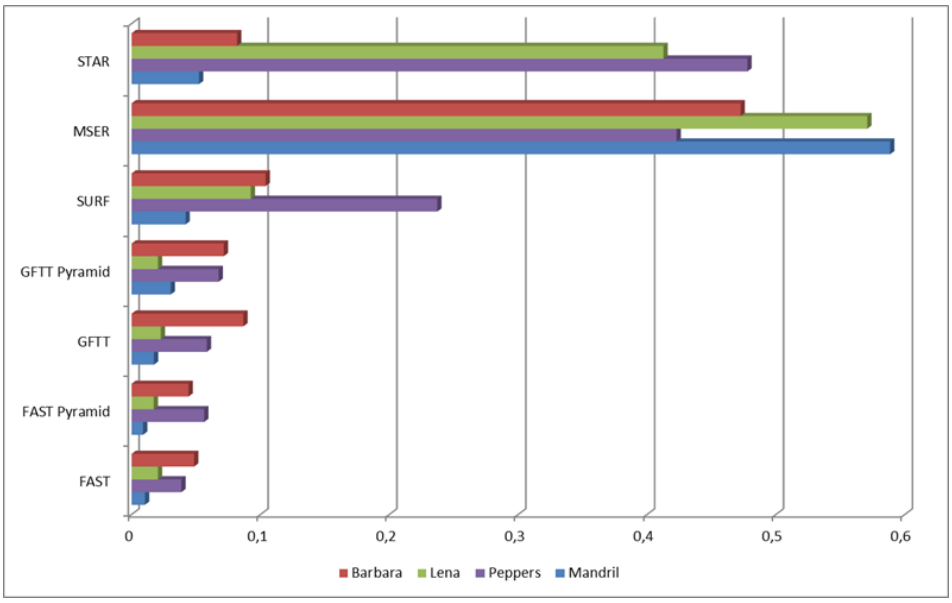
K detekování zájmových bodů jsem se kvůli rychlosti detekce rozhodl použít funkce nabízené knihovnou OpenCV. K výběru vhodného detekčního algoritmu jsem jako zdroj informací použil [3]. Eugene Khvedchenya zde srovnává algoritmy poskytované knihovnou OpenCV mezi něž patří detektory STAR, MSER, SURF a další.

Volbu detekčního algoritmu jsem založil hlavně na dvou faktorech. Tím prvním je časová náročnost detekování jednoho zájmového bodu, viz obrázek 7. Druhým faktorem je pak rozdíl pozice detekovaného bodu v obrázcích s mírně změněnou světlostí. Odpovídající graf je vidět na obrázku 8.

Po tomto srovnání jsem se nejdříve rozhodl využít FAST detektor, kvůli jeho rychlosti detekce. Vlastním testováním detektorů jsem ale přistoupil na GFTT detektor, protože FAST detektor má tendenci detekovat spíše více kontrastní hrany, než méně kontrastní rohy. Proto jsem tedy volil GFTT detektor, jelikož ten pro detekci zájmových bodů využívá Harrisův algoritmus. Tímto jsem získal při detekci zájmových bodů body, odpovídající rohům objektů v obraze, a bylo tak možno přesněji tyto body v obraze sledovat napříč dalšími snímky.



Obrázek 7: Časová náročnost detekce jednoho zájmového bodu v milisekundách



Obrázek 8: Posun bodu v závislosti na mírné změně světlosti obrazu

Pro další zpřesnění bodů jsem pak použil funkci cornerSubPix, která iterativně na základě subpixelu zpřesňuje pozici zájmových bodů blíže rohům v obraze.

Ukázku detekovaných zájmových bodů v obraze pomocí GFTT detektoru se subpixelovým zpřesněním lze vidět na obrázku 9.

Obrázek 9: Detekované zájmové body.

4.5 Sledování zájmových bodů

Pro sledování, resp. párování zájmových bodů se obvykle používá tzv. matching, tedy spárování nejvíce si odpovídajících bodů rozpoznaných nezávisle na sobě v obou obrazech. Takový způsob se však hodí spíše na stereoskopický systém, který má pravidla, díky kterým, se na spárování můžeme dostatečně spolehnout. Mezi tyto pravidla patří např. odpovídající si velikost objektů (na žádaném okolí detekovaného zájmového bodu) na obou snímcích, nebo také ten fakt, že párovaná okolí bodů lze hledat v odpovídajících si vodorovných hladinách a jejich blízkém okolí.

Na podobné věci jsem se však u monoskopického systému spolehnout nemohl, i přesto jsem však vyzkoušel několik testovacích pokusů s tímto spárováním zájmových bodů. Bohužel se jako dostatečně spolehlivý neprokázal ani RobustMatcher s RANSAC algoritmem.

A tak jsem přistoupil k takovému způsobu sledování zájmových bodů, kdy se zájmové body rozpoznají jen na jednom obraze a na následujícím se vyhledá jejich nejvíce odpovídající dvojník se stejným okolím zájmového bodu. Na tento způsob sledování se mi osvědčilo použít funkci knihovny OpenCV s názvem calcOpticalFlowPyrLK. Tato funkce počítá optický tok (optical flow) vstupních bodů mezi dvěma snímky použitím pyramidové Lucas-Kanade metody popsané v kapitole 3.3. Tímto způsobem jsem získal poměrně přesnou polohu obrazů zájmových bodů na druhém snímku.

Tento způsob sledování jsem pak opakoval v závislosti na rychlosti jízdy vozidla, resp. rychlosti snímání tak, aby byl optický posun zájmových bodů na snímcích znatelný. To proto, že čím větší je vzdálenost pozice kamery prvního snímku a pozice kamery posledního rozpoznávaného snímku, tím přesnější je následný výpočet fundamentální matice.

Snímek, na kterém proběhla detekce zájmových bodů budu dále nazývat jako "první snímek", a snímek, kterým končí iterace sledování zájmových bodů budu nazývat "poslední snímek". A to i přesto, že před prvním snímkem a za posledním snímkem může videosekvence obsahovat další nezpracovávané snímky.

Na obrázku 10 je vidět spojnice mezi zájmovými body prvního a posledního sledovaného snímku.

Obrázek 10: Odpovídající si body na 2 snímcích.

4.6 Fundamentální a esenciální matice

Pro výpočet fundamentální matice v mé implementaci jsem použil funkci findFundamentalMat knihovny OpenCV, kterou jsem pomocí parametrů přinutil vypočítávat fundamentální matici pomocí 8-bodového algoritmu (8-point algorithm), který vyžaduje 8 a více korespondujících bodů. Každý takový pár je pak brán jako podmínka a měl by

splňovat rovnici

$$pt_2^{iT} \cdot F \cdot pt_1^i = 0 \tag{21}$$

resp.

$$\left[x_2^i x_1^i, x_2^i y_1^i, x_2^i, y_2^i x_1^i, y_2^i y_1^i, y_2^i, x_1^i, y_1^i, 1\right] \cdot [f_{11}, f_{12}, f_{13}, f_{21}, f_{22}, f_{23}, f_{31}, f_{32}, f_{33}]^T = 0 \tag{22}$$

S $N \geq 8$ páry pak vznikne lineární systém

$$A \cdot \mathbf{f} = 0 \tag{23}$$

kde i-tý řádek matice A odpovídá i-tému korespondujícímu páru zapsanému 9-prvkovým vektorem, jak je uvedeno výše.

Výsledkem pak je vlastní vektor odpovídající nejmenšímu vlastnímu číslu matice $A^T A$, který je následně přepsán do fundamentální čtvercové matice.

Odpovídající si epipolární linie získané na základě takto vypočtené fundamentální matice jsou vidět na obrázku 11.

Fundamentální matice musí být co nejpřesnější, proto jsem při detekci zájmových bodů volil detekování menšího množství bodů, které jsou však díky odpovídajícím rohům v obraze dobře sledovatelné a tím i dostatečně přesné.

Abych mohl později získat vzájemnou rotaci a translaci mezi snímky bylo potřeba z fundamentální matice F a vnitřní matice kamery K vypočítat esenciální matici E, a to pomocí následujícího vzorce:

$$E = K^T \cdot F \cdot K \tag{24}$$

Esenciální matice však má mít hodnost 2, proto je potřeba i toto ošetřit. Zajistil jsem to tak, že jsem esenciální matici E rozložil pomocí singulárního rozkladu:

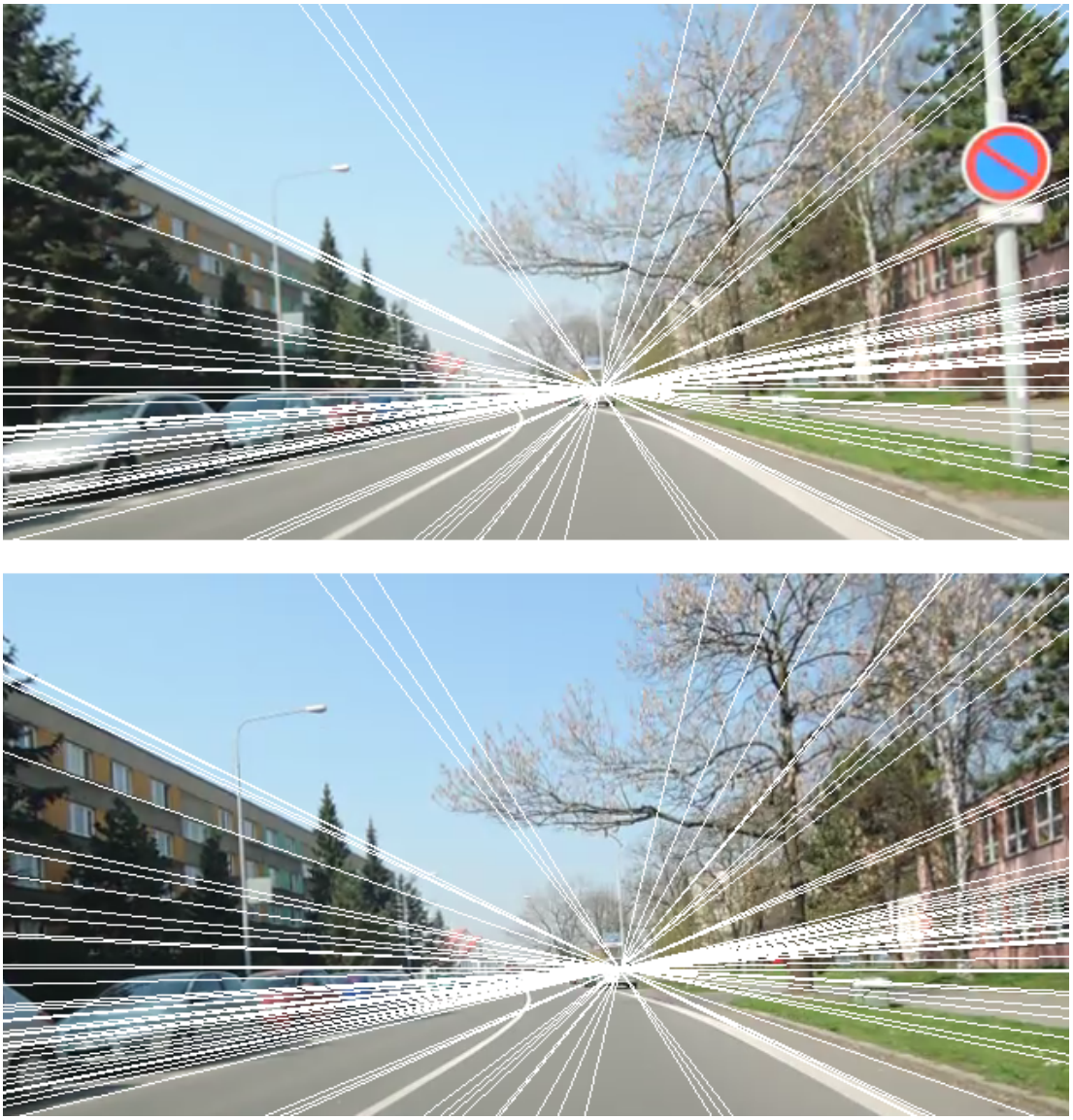
$$U \cdot D_0 \cdot V^T = E \tag{25}$$

a poté opět složil za pomocí diagonální matice D:

$$E = U \cdot D \cdot V^T \tag{26}$$

kde

$$D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{27}$$



Obrázek 11: Korespondující epipolární linie.

4.7 Transformační matice kamer

Pro výpočet transformačních matic kamer jsem potřeboval nejprve vypočítat rotační matici a translační vektor z esenciální matice. Jak je ale vidno na obrázku 5, existují 4 způsoby jak určit vzájemnou pozici kamer a jen jeden z nich je správný.

Bohužel však neexistuje způsob, jak zjistit ještě před triangulací, který případ je pro danou esenciální matici ten správný a tak musí výpočet proběhnout pro všechny 4 případy

a na základě rekonstruovaných bodů se rozhodnout, který je správný. V následujících krocích popíšu, jak jsem analyzoval správný případ vzájemné pozice kamer:

1. Opětovným rozkladem esenciální matice jsem získal matice U a V:

$$E = U \cdot D \cdot V^T. \tag{28}$$

2. Transformační matice první kamery jsem definoval jako:

$$P_1 = (\ I \mid 0\ ). \tag{29}$$

3. (a) Transformační matice druhé kamery pak jako jeden z případů:

$$P_2 = (U \cdot W \cdot V^T \mid U \cdot (0,0,1)^T) \tag{30}$$

$$P_2 = (U \cdot W \cdot V^T \mid -U \cdot (0,0,1)^T) \tag{31}$$

$$P_2 = (U \cdot W^T \cdot V^T \mid U \cdot (0,0,1)^T) \tag{32}$$

$$P_2 = (U \cdot W^T \cdot V^T \mid -U \cdot (0,0,1)^T). \tag{33}$$

   (b) Následně jsem provedl triangulaci zájmových bodů popsanou v kapitole 4.8.

   (c) Triangulované body jsem pak otestoval, zda se nacházejí před oběma kamerama.

4. Porovnáním počtu triangulovaných bodů před kamerou z kroku 3 jsem zjistil, který případ transformační matice druhé kamery nejvíce odpovídá skutečnosti.

4.8 Triangulace

Triangulací zájmových bodů detekovaných v kapitole 4.4 bych získal jen mírnou informaci o tom, jak scéna prostorově vypadá, protože jsem detekoval zájmových bodů poměrně málo, tj. v řádu desítek. Na pozdější výpočet fundamentální matice z těchto bodů však postačovalo, či dokonce bylo vyžadováno méně bodů vyšší kvality (tj. body, které mají potenciál být přesněji sledovatelné) než opačně.

Pro triangulaci je však žádoucí, dostat co největší ponětí o scéně a bylo tedy zapotřebí detekovat více bodů. Abych se v sekvenci analyzovaných snímků nevracel nazpátek, paralelně s detekcí zájmových bodů jsem detekoval ve stejném snímku také body určené k triangulaci. Těchto bodů by mělo být co nejvíce a nemusí být nutně jejich pozice dána

pevnými rohy v obraze. Proto jsem pro detekci těchto bodů využil FAST detektor. Ten za velmi malou cenu času, v porovnání s jinými detektory zajistí rozpoznání velkého počtu víceméně sledovatelných bodů.

Stejně jako jsem paralelně provedl detekce, provedl jsem i sledování těchto bodů paralelně se sledováním zájmových bodů. Na konci tohoto procesu mám pak pole dvojic odpovídajících si bodů z prvního a posledního sledovaného snímku, které budou sloužit k následné triangulaci.

Tyto dvojice jsem pak otestoval na dostatečnou přesnost rovnicí (34).

$$\left|pt_2^T \cdot F \cdot pt_1\right| < 1.0 \tag{34}$$

kde

- pt_1 a pt_2 označují korespondující bodu na prvním a posledním snímku,
- F je dříve vypočtená fundamentální matice.

V ideálním případě dvojice bodů odpovídá rovnici (17).

K triangulaci jsem použil algoritmus lineární triangulace nejmenších čtverců popsaný v [7]. Tuto triangulaci už dříve implementoval Roy Shil a publikoval, viz [11], a tak jsem ji použil s obměnou kódu pro OpenCV 2.4.8.

Algoritmus lineární triangulace nejmenších čtverců pracuje tak, že nejdříve pro každou dvojici korespondujících bodů vypočítá matici A o rozměru 4x3:

$$A = \begin{bmatrix} u_0(x) \cdot P_0(2,0) - P_0(0,0) & u_0(x) \cdot P_0(2,1) - P_0(0,1) & u_0(x) \cdot P_0(2,2) - P_0(0,2) \\ u_0(y) \cdot P_0(2,0) - P_0(1,0) & u_0(y) \cdot P_0(2,1) - P_0(1,1) & u_0(y) \cdot P_0(2,2) - P_0(1,2) \\ u_1(x) \cdot P_1(2,0) - P_1(0,0) & u_1(x) \cdot P_1(2,1) - P_1(0,1) & u_1(x) \cdot P_1(2,2) - P_1(0,2) \\ u_1(y) \cdot P_1(2,0) - P_1(1,0) & u_1(y) \cdot P_1(2,1) - P_1(1,1) & u_1(y) \cdot P_1(2,2) - P_1(1,2) \end{bmatrix} \tag{35}$$

a vektor b:

$$b = \begin{bmatrix} -u_0(x) \cdot P_0(2,3) - P_0(0,3) \\ -u_0(y) \cdot P_0(2,3) - P_0(1,0) \\ -u_1(x) \cdot P_1(2,3) - P_1(0,3) \\ -u_1(y) \cdot P_1(2,3) - P_1(1,3) \end{bmatrix} \tag{36}$$

kde

- P_0 a P_1 jsou transformační matice kamer,

- u_0 a u_1 jsou korespondující body v prvním a druhém snímku,
- a $L(m, n)$ odpovídá prvku matice L na (m+1). řádku a na (n+1). sloupci.

A následně vypočte vektor x určený takto:

$$A \cdot x = b \tag{37}$$

Jednotlivé složky trojprvkového vektoru x pak odpovídají x-ové, y-ové a z-ové souřadnici vypočteného bodu.

Po skončení výpočtu je pak na výstupu tohoto algoritmu odpovídající počet 3D bodů.

4.9 Pozice kamer

Pozice kamer v prostoru je možné vypočítat z transformačních matic kamer. Já se však rozhodl vypočítávat jejich pozici zpětně z vypočtených 3D bodů. Tím získám jednak přesnější vzájemnou pozici kamer a bodů, ale také je z takto vypočtených pozic snadné zjistit, zda byl výpočet triangulace dostatečně přesný.

Zdrojový kód výpočtu zpětné pozice kamery z 3D bodů je vidět ve výpisu 1, kde

- points3Df jsou rekonstruované 3D body,
- points2D jsou odpovídající body na snímku dané kamery,
- distortion je matice obsahující parametry zkreslení obrazu dané kamery,
- funkce solvePnPRansac slouží pro určení pozice objektu ze zadaných bodů za použití RANSAC algoritmu,
- funkce Rodrigues pak převádí rotační vektor na rotační matici.

```
cv::Point3d cameraPosition( cv::vector<cv::Point3d> points3D,
                            cv::vector<cv::Point2f> points2D,
                            cv::Mat K,
                            cv::Mat distortion)
{
    cv::Mat tvec, rvec;
    std::vector<cv::Point3f> points3Df;
    for ( unsigned int i = 0; i < points3D.size(); i++)
    {
```

```
        points3Df.push_back((cv::Point3f)points3D[i]) ;
    }
    cv::solvePnPRansac(points3Df, points2D, K, distortion, rvec, tvec, false);
    cv::Mat rMat;
    cv::Rodrigues(rvec, rMat);
    cv::Mat camPoseMat(-(rMat.t()) * tvec);
    cv::Point3d camPose = cv::Point3d(camPoseMat.at<double>(0,0), camPoseMat.at<double
        >(1,0), camPoseMat.at<double>(2,0));
    return camPose;
}
```

Výpis 1: Výpočet pozice kamery

Na základě vypočtených pozic kamer pak zakládám zjištění na kolik byla rekonstrukce úspěšná. Při stoprocentní úspěšnosti totiž platí tyto pravidla:

- pozice první kamery odpovídá počátečnímu bodu, tedy [0, 0, 0],
- vzdálenost obou kamer je rovna 1.

Právě na druhém pravidle jsem založil rozhodnutí, zda se proces rekonstrukce včetně nalezení fundamentální matice bude opakovat, nebo zda je dostatečně úspěšný.

4.10 Vizualizace

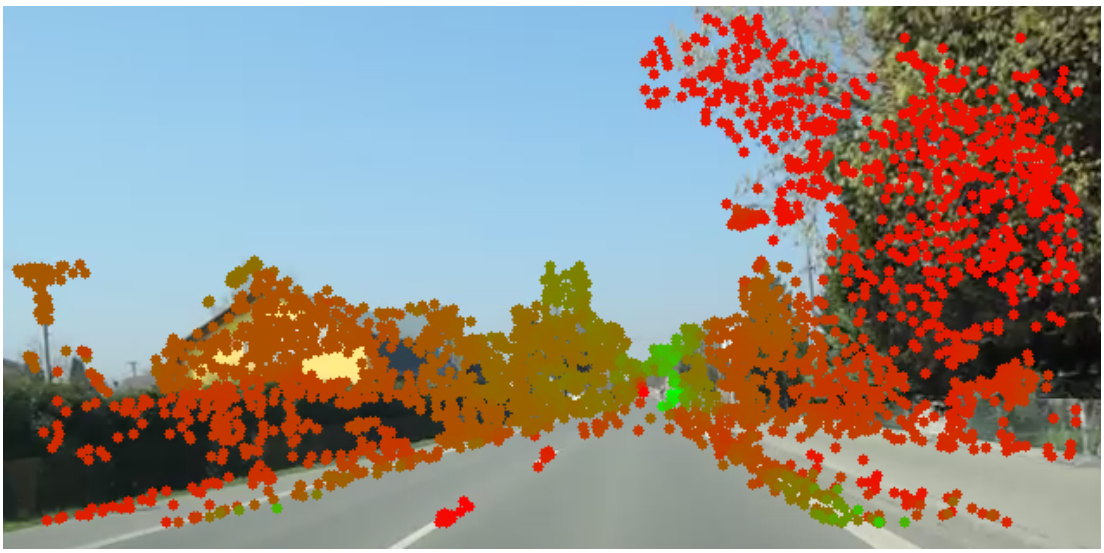
Aby bylo zjevné, nakolik je výsledná rekonstrukce přesná, je potřeba rekonstruované body nějak vizualizovat, a tak jsem zvolil tato zobrazení:

- vykreslení bodů přímo do snímku pomocí barevného gradientu červená-zelená,
- 3D vizualizace pomocí PCL knihovny.

4.10.1 Barevná z-mapa

Pro zobrazení pomocí barevného gradientu simulujícím hloubkovou mapu jsem se rozhodl proto, že je z výsledku jasně vidět vzdálenost rekonstruovaných bodů od kamery. Toto zobrazení je ideální pro situace, kdy je potřeba ihned analyzovat viditelný obraz. Takovou situací může být upozornění řidiče na blížící se vozidlo, pomoc při parkování, apod.

Výsledný obraz pak vypadá např. jako na obrázku 12. Čím blíže jsou rekonstruované body blíže kameře, tím více jsou jejich obrazy zabarveny do červena, naopak čím jsou vzdálenější, tím více jsou zbarvené zeleně.

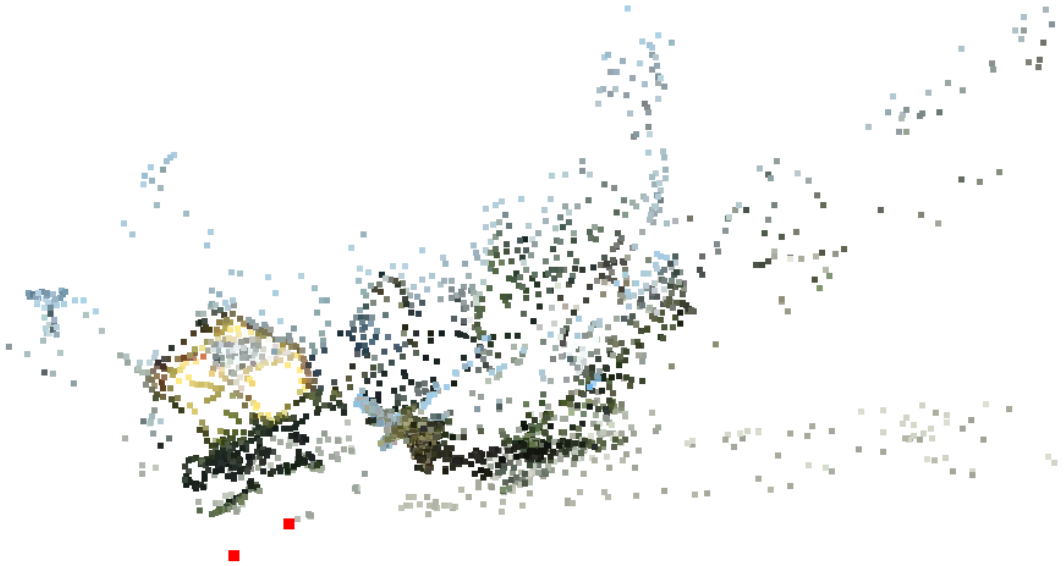


Obrázek 12: Ukázka vyobrazení bodů pomocí z-mapy.

4.10.2 3D vizualizace

Abych si mohl vypočtené body lépe prohlédnou a analyzovat výslednou rekonstrukci, bylo nutné, abych pro vizualizaci rekonstruovaných bodů zvolil také nějakou interaktivní metodu zobrazení. Jelikož knihovna OpenCV ve verzi 2.4.8 nemá k 3D vizualizaci potřebné nástroje a funkce, musel jsem se poohlédnout jinam.

Po prostudování různých řešení jsem nakonec zvolil knihovnu PCL, která nabízí velké množství nástrojů pro vyobrazení 3D bodů a jejich interaktivitu při prohlížení. Výsledná vizualizace sice na obrázku 13 není vhodná k analyzování rekonstruovaných bodů ani k získání prostorového vjemu, v prostředí PCL vizualizéru je však díky interaktivitě dostatečně přehledným způsobem vyobrazení.



Obrázek 13: Ukázka vizualizace bodů pomocí PCL vizualizéru.

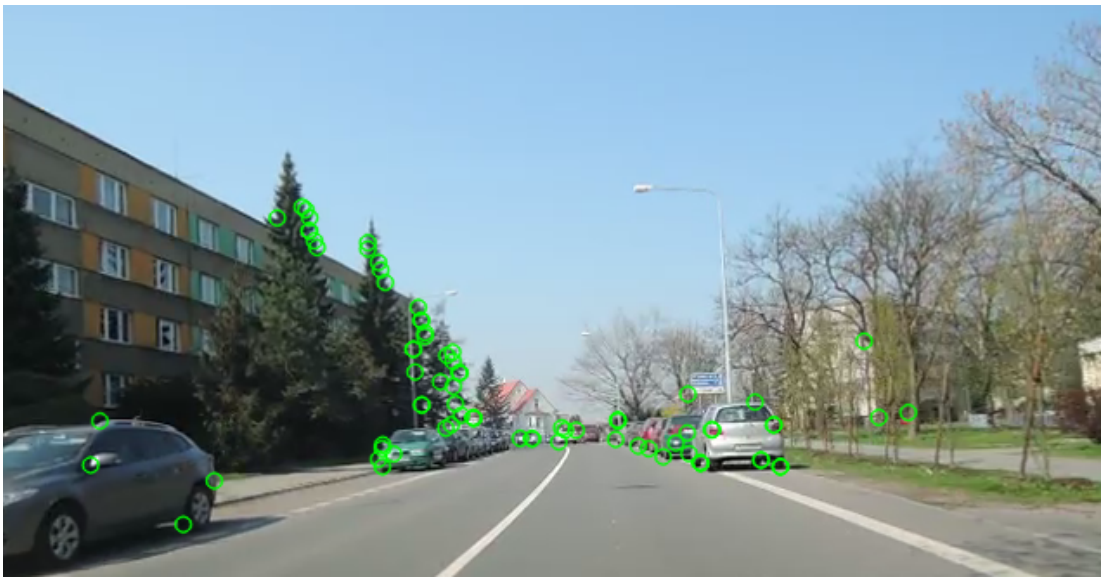
5 Diskuze a výsledky

Pro analýzu výsledné rekonstrukce jsem zvolil dvě scény lišící se místem pořízení a způsobem, jak byla scéna natáčena. První scénou je okolí městské ulice, viz obrázek 14. Tuto videosekvenci jsem pořídil z jedoucího auta. Druhou scénou pak je přílesní horská cesta, viz obrázek 15, která byla natáčena ručně při chůzi.

5.1 Průběh rekonstrukce

V první scéně se pro výpočet fundamentální matice generovalo GFTT detektorem na prvním snímku předepsaných 100 zájmových bodů, jejichž počet v průběhu sledování přes 10 snímků klesl na 83 bodů, viz obrázek 14. Příčinou úbytku počtu bodů je jednak přesun bodu mimo obraz a jednak nenalezení dostatečně odpovídající oblasti okolí bodu v následujícím snímku.

V druhé scéně je pak počet detekovaných bodů redukován ze stejných příčin ze 100 na 58 zájmových bodů, viz obrázek 15. Počet snímků, na kterých probíhalo sledování zájmových bodů, byl však v tomto případě kvůli pomalé chůzi při natáčení videosekvence navýšen na 40 snímků, aby byla vzdálenost kamery při záznamu prvního a posledního snímku dostatečně znatelná, což je důležité pro výpočet fundamentální matice.



Obrázek 14: Snímek městské ulice s vykreslenými sledovanými zájmovými body

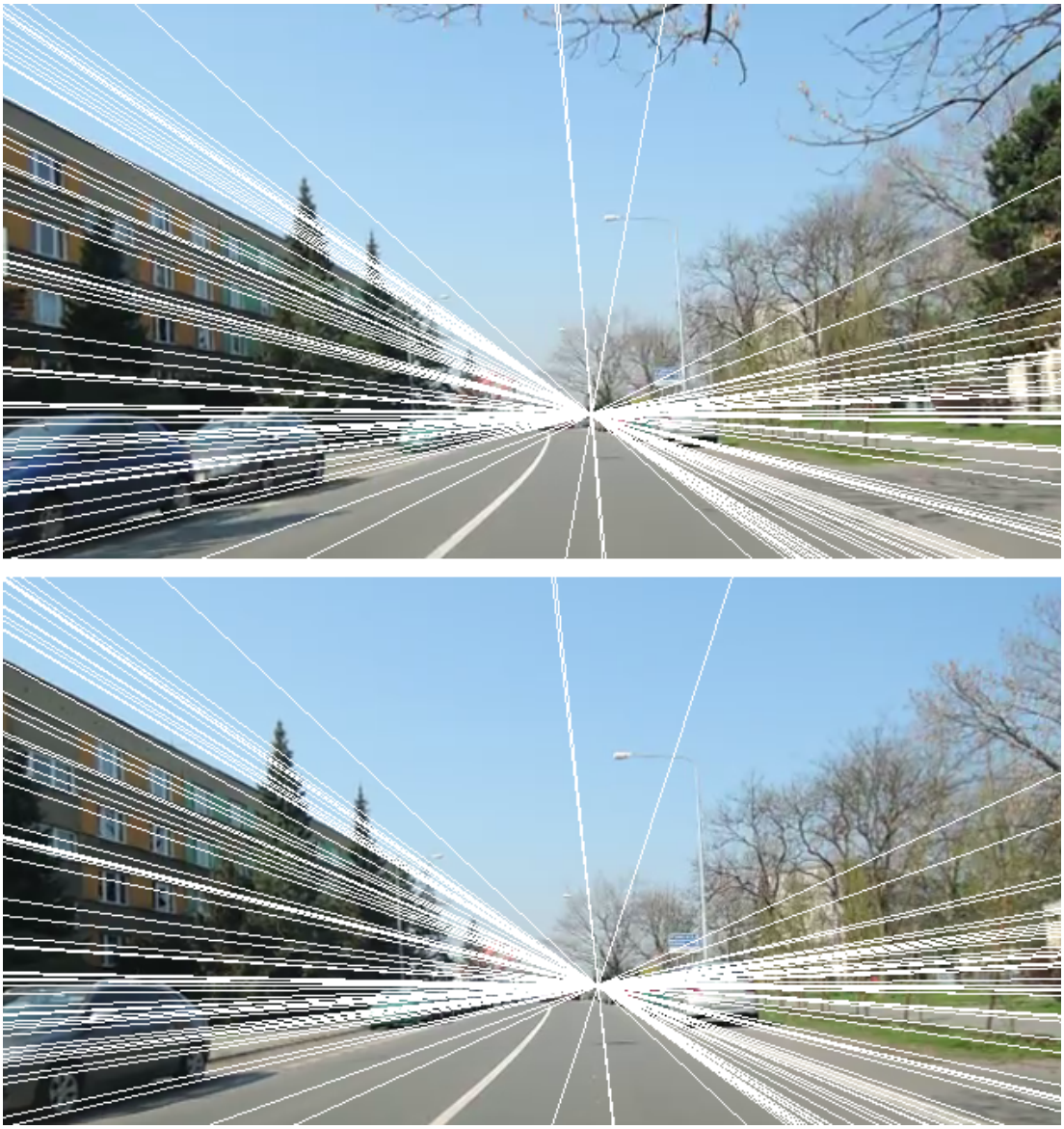
Obrázek 15: Snímek přílesní cesty s vykreslenými sledovanými zájmovými body

Následoval výpočet fundamentální matice, jejíž přesnost se dá určit konkrétněji až s dalšími kroky. Jistým ukazatelem však můžou být párované epipolární linie pro každou ze scén. Pro městskou ulici na obrázku 16, pro přílesní cestu pak na obrázku 17. V obou těchto případech je vidět, že epipolární linie si poměrně přesně odpovídají a tak bylo možné očekávat, že fundamentální matice jsou vypočteny dostatečně přesně. Bohužel tomu tak není pokaždé.

Přibližně v polovině případů testovaných scén byla po sledování zájmových bodů vypočtena fundamentální matice špatně, což bylo pozorovatelné neodpovídajícími si epipolárními liniemi. Bylo to tehdy, kdy bylo při detekci zájmových bodů detekováno více bodů, které neodráží pevný bod v prostoru. Tyto body pak při svém vyšším počtu ovlivní negativně výpočet fundamentální matice.

Mezi takové body patří body na rozhraní bližšího a vzdálenějšího objektu. Takový bod pak způsobuje, že jeho sledování má tendenci se přizpůsobovat oběma objektů. Příklad takového bodu je vidět na obrázku 18.

Druhým typem bodů, které způsobují nepřesný výpočet fundamentální matice jsou body, které sice odpovídajících bodům na povrchu konkrétních objektů, ale jedná se pohybující se objekty. Typickým příkladem takových bodů jsou detekované body na protijedoucím vozidle, či na po chodníku procházejícím se chodci.



Obrázek 16: Odpovídající si epipolárních linie na první scéně

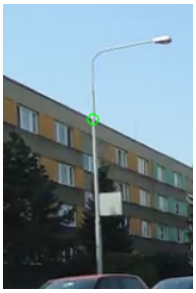
V ukázkových příkladech však tato situace nenastala a výpočet fundamentálních matic tak proběhl s dostatečnou přesností.

Po vypočtení fundamentální matice následuje zaměření se na samotnou rekonstrukci scény. Jelikož je pro ni nutno řádově vyšší počet bodů, proběhla paralelně s detekcí a sledováním zájmových bodů i detekce a sledování bodů určených k rekonstrukci.

Počet bodů v městské scéně se od detekce po poslední krok sledování zredukoval z

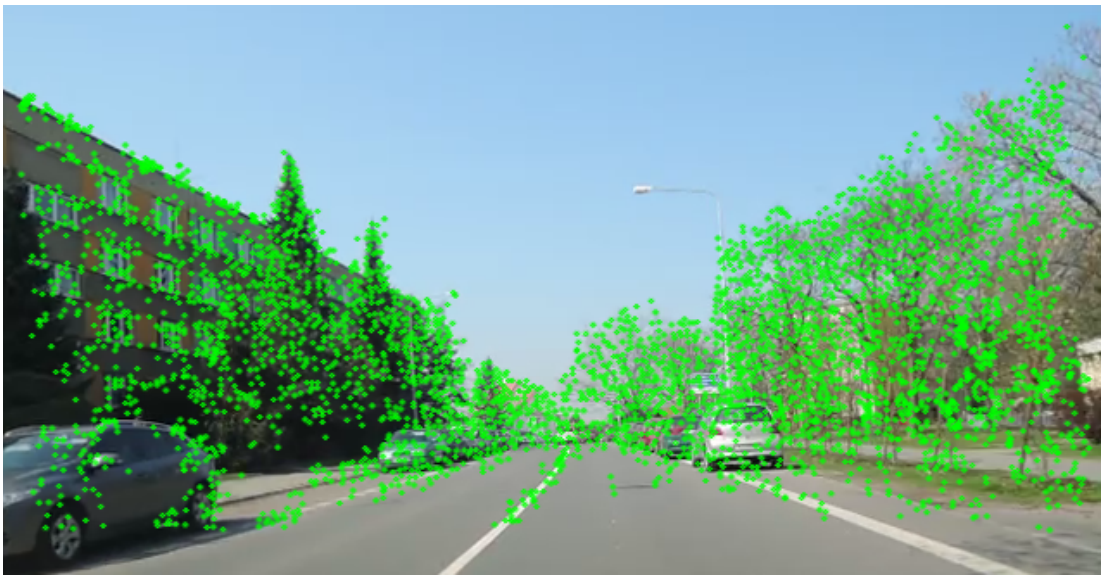
Obrázek 17: Odpovídající si epipolárních linie na scéně druhé

6 220 na 4 173 body. Tento počet byl následně zredukován na 3 997 bodů. Vyřazeny byly totiž ty body, jež neodpovídaly rovnici (34). Stejně tak byl počet k rekonstrukci určených bodů u přílesní scény redukován z původních 13 723 na 8 243 body. Tyto body jsou vidět na obrázcích 19 a 20. Na druhém z obrázků je vidět, jak pomáhá detekci a sledování „tex-



Obrázek 18: Ukázka bodu neodrážejícího pevný bod v prostoru

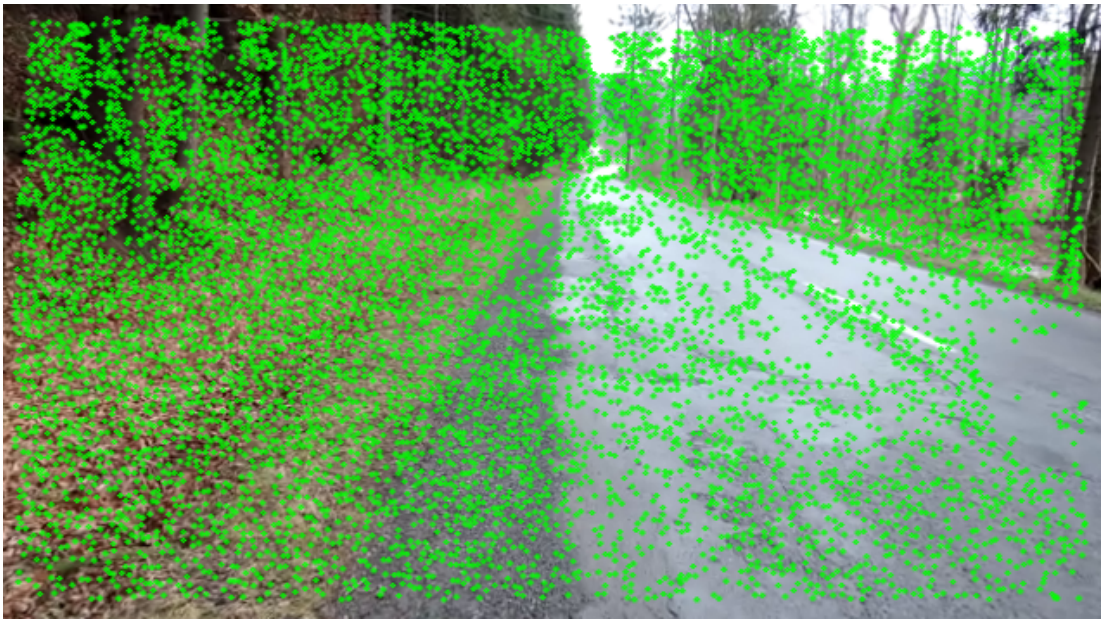
turovaný“ povrch scény. Oproti městské scéně tak body určené k rekonstrukci pokrývají téměř celý snímek, zatímco u této městské scény je vidět nemožnost správné detekce a sledování bodů zejména v oblasti silnice a oblohy.



Obrázek 19: Detekované a sledované body městské scény

Na obrázcích 21 a 22 je pak vidět posun sledovaných bodů vzhledem k obrazu, odrážející tzv. optický tok mezi prvním a posledním snímkem. Zeleně jsou zabarveny čáry, které náleží dvojici bodů, jež odpovídají rovnici (34), naopak červeně jsou zabarveny čáry, které této rovnici neodpovídají.

Po tomto kroku rekonstrukce následovalo generování 4 případů rotační matice a translačního vektoru pro výpočet transformačních matic kamer, viz kapitola 4.7.



Obrázek 20: Detekované a sledované body přílesní scény

Pro každý z případů byla provedena triangulace bodů určených k rekonstrukci a následně ověřeno, kolik rekonstruovaných bodů se nachází před oběma kamerami.

Počty bodů před oběma kamerami pro každý ze 4 případů jsou pro obě scény uvedeny v Tabulkách 1 a 2.

Na základě těchto tabulek byl pro výpočet transformačních matic kamer zvolen u městské scény případ č.3 a u přílesní scény případ č.4. K oběma scénám pak byla vybrána i odpovídající, dříve provedená triangulace rekonstruovaných bodů.

Po tomto kroku následoval zpětný výpočet kamer.

Kvalita rekonstrukce se dá nejlépe posoudit vizuálním pohledem na rekonstruované body ať už ve formě hloubkové mapy, či 3D vizualizace. Bylo však nutné, abych použil nějaký ukazatel, který by rozhodoval o tom, zda je výsledná rekonstrukce dostatečně přesná a v opačném případě provedl opětovnou rekonstrukci z dalších snímků videosekvence.

Na základě experimentů jsem pak zjistil, že mi jako poměrně slušný ukazatel přesnosti rekonstrukce posloužila hodnota vzájemné vzdálenosti zpětně vypočtených kamer. Čím blíže se vzdálenost těchto kamer blížila 1, tím přesněji výsledná rekonstrukce odpovídala snímané scéně.

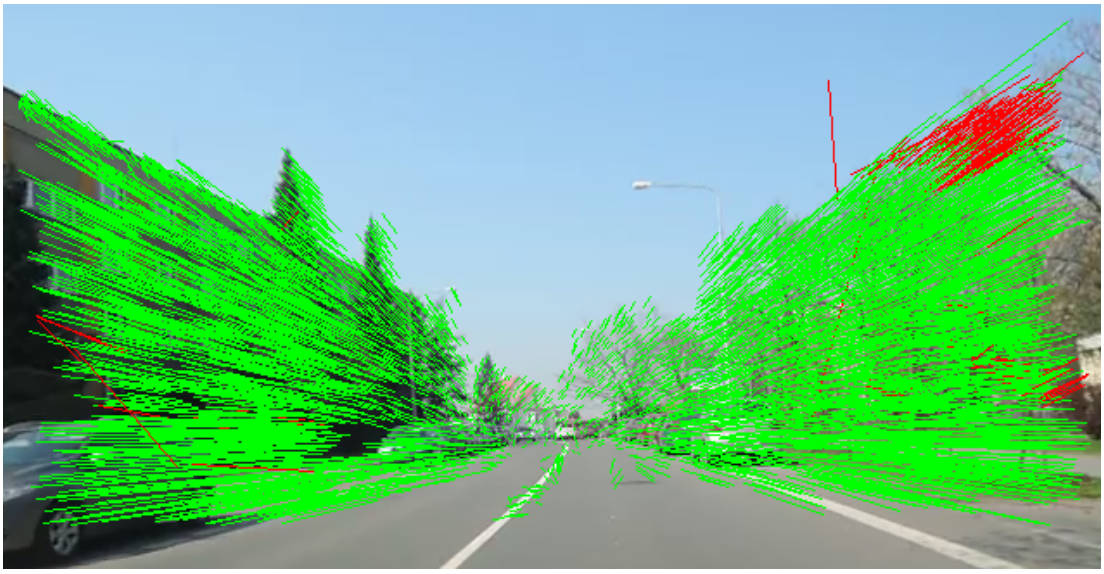
—	Před kamerami	Celkový počet
Případ 1	23	3997
Případ 2	0	3997
Případ 3	3973	3997
Případ 4	1	3997

Tabulka 1: Počet bodů před kamerami ve městské scéně.

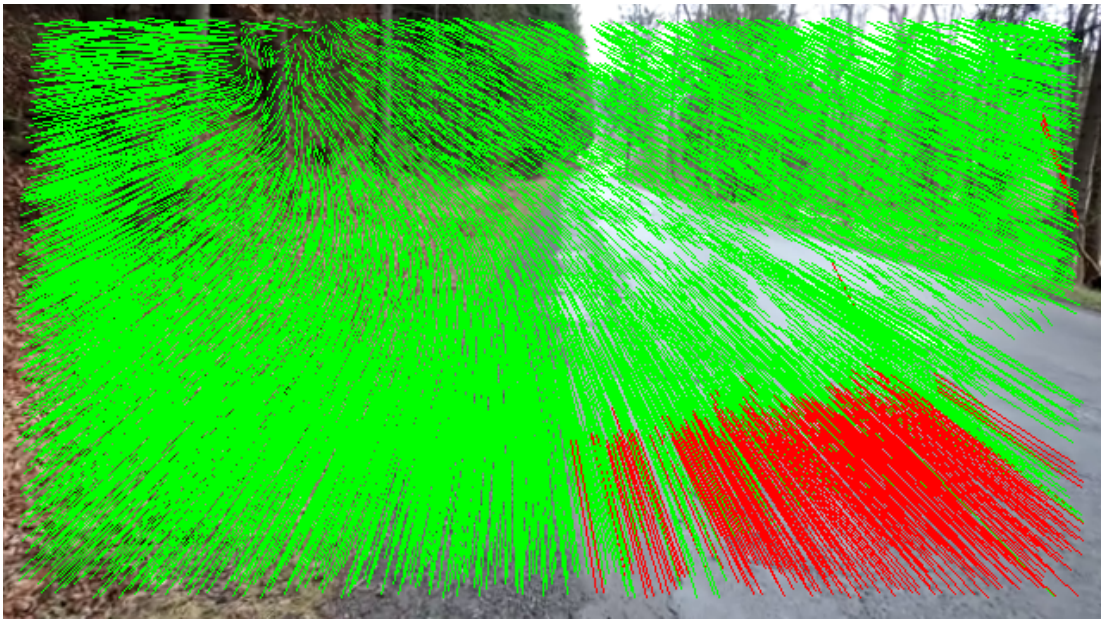
—	Před kamerami	Celkový počet
Případ 1	0	8243
Případ 2	0	8243
Případ 3	0	8243
Případ 4	8214	8243

Tabulka 2: Počet bodů před kamerami v přílesní scéně.

Hodnota vzdálenosti kamer při rekonstrukci městské scény byla rovna 0.982133, v přílesní scéně pak byla vzdálenost kamer rovna 1.03436.



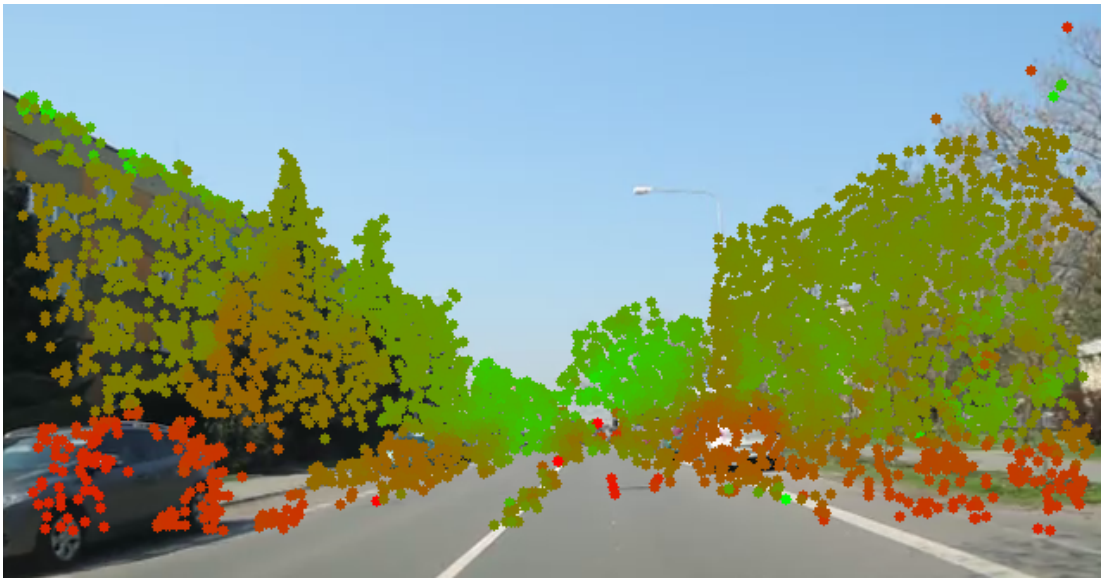
Obrázek 21: Optický tok městské scény



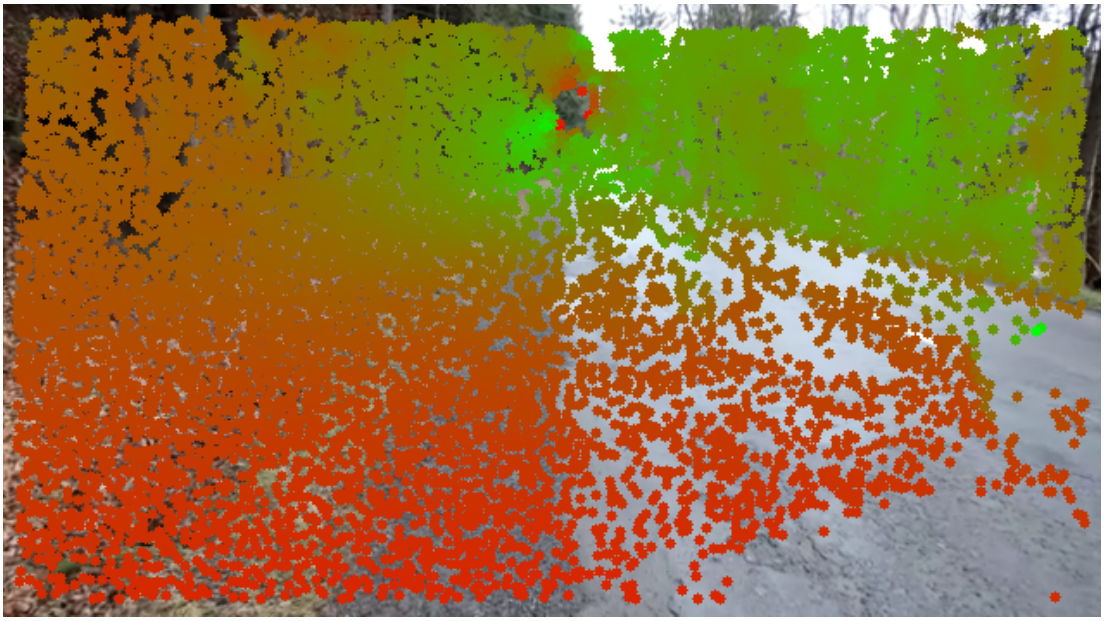
Obrázek 22: Optický tok přílesní scény

5.2 Diskuze vizualizovaných výsledků

Vizualizace rekonstruovaných bodů obou scén pomocí barevné hloubkové mapy je vidět na obrázcích 23 a 24.



Obrázek 23: Hloubková mapa městské scény



Obrázek 24: Hloubková mapa přílesní scény

Na obrázku 23 je vidět, že většina zobrazované scény prostorově odpovídá rekonstruované scéně, obsahuje však nepřesné oblasti.

Jednou z takových oblastí je ve spodní střední části snímku oblast odpovídající okolí epipólu, tedy bodu, kde se epipolární linie střetávají. Takové body se ve snímku objevují v případě, když se epipól nachází na snímku. Důvodem nepřesnosti je fakt, že čím blíže se obraz rekonstruovaného bodu na snímku blíží epipólu, tím větší jsou kladeny nároky na jeho přesnost. Vyvarovat se takovýchto nepřesností je možné například tím, že oblasti okolo epipólu budu už v počátcích vyřazovat z rekonstrukce. Stejnou oblast nepřesně rekonstruovaných bodů je vidět i na obrázku 24 v horní střední části snímku.

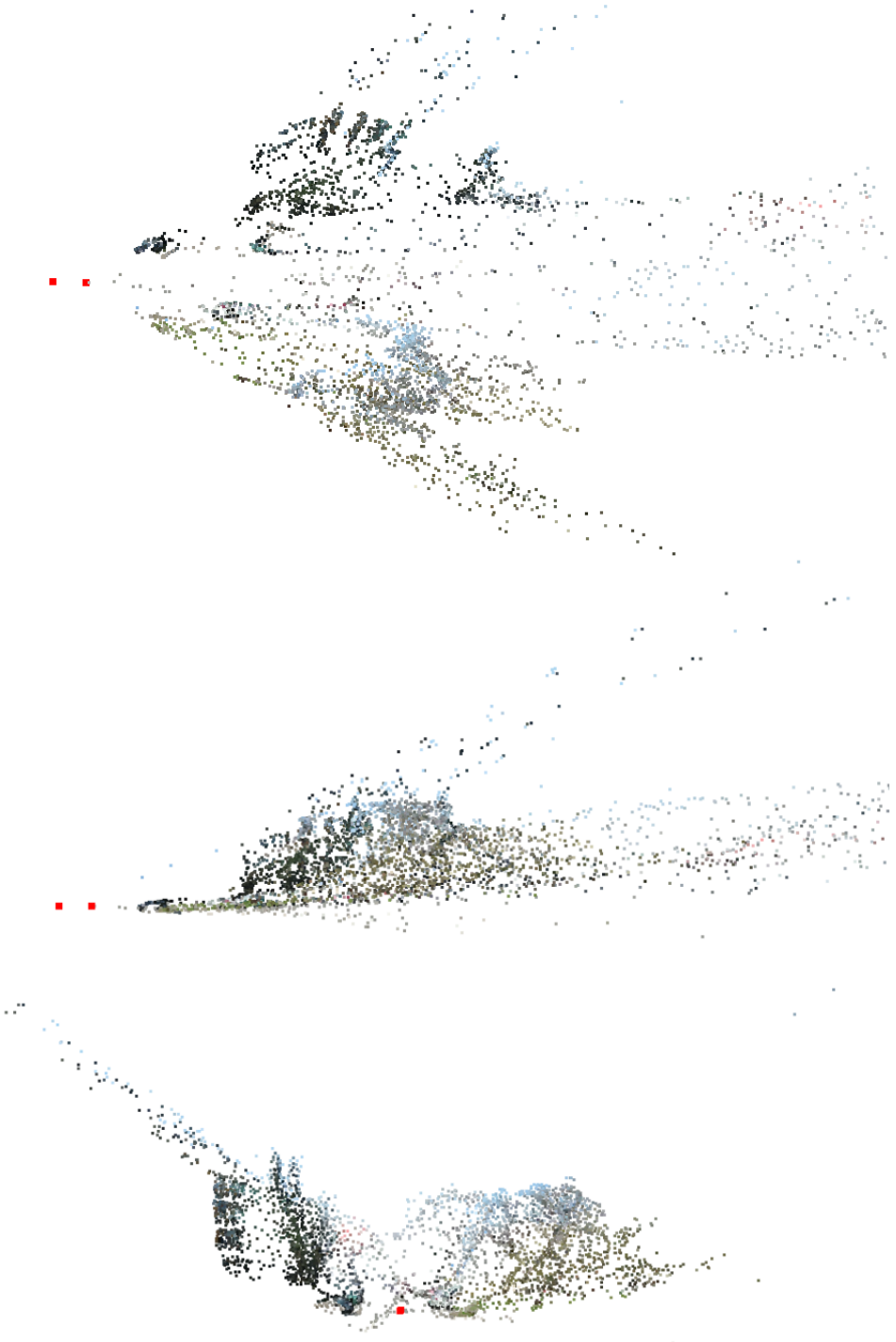
Další oblast nepřesně rekonstruovaných bodů je oblast okolo krajnice a středové čáry silnice. Tyto oblasti lze pozorovat na obou obrázcích 23 a 24. Důvod špatné rekonstrukce těchto bodů je nutné hledat už v počátcích rekonstrukce a to při sledování bodů po detekci. Tyto body totiž nemají žádný pořádně záchytný fragment v obraze, který by jim napověděl pozici dalšího pohybu při sledování. Proto mají tendenci setrvávat v průběhu sledování stále na přibližně stejné pozici v obraze, hlavně v případě, kdy se trajektorie kamery vzhledem k zakřivení silnice mění jen velmi málo.

Mezi neurčité body setrvávající na přibližně stejném místě je pak možné zařadit i body sledované na horním okraji budovy v levé horní části obrázku 24.

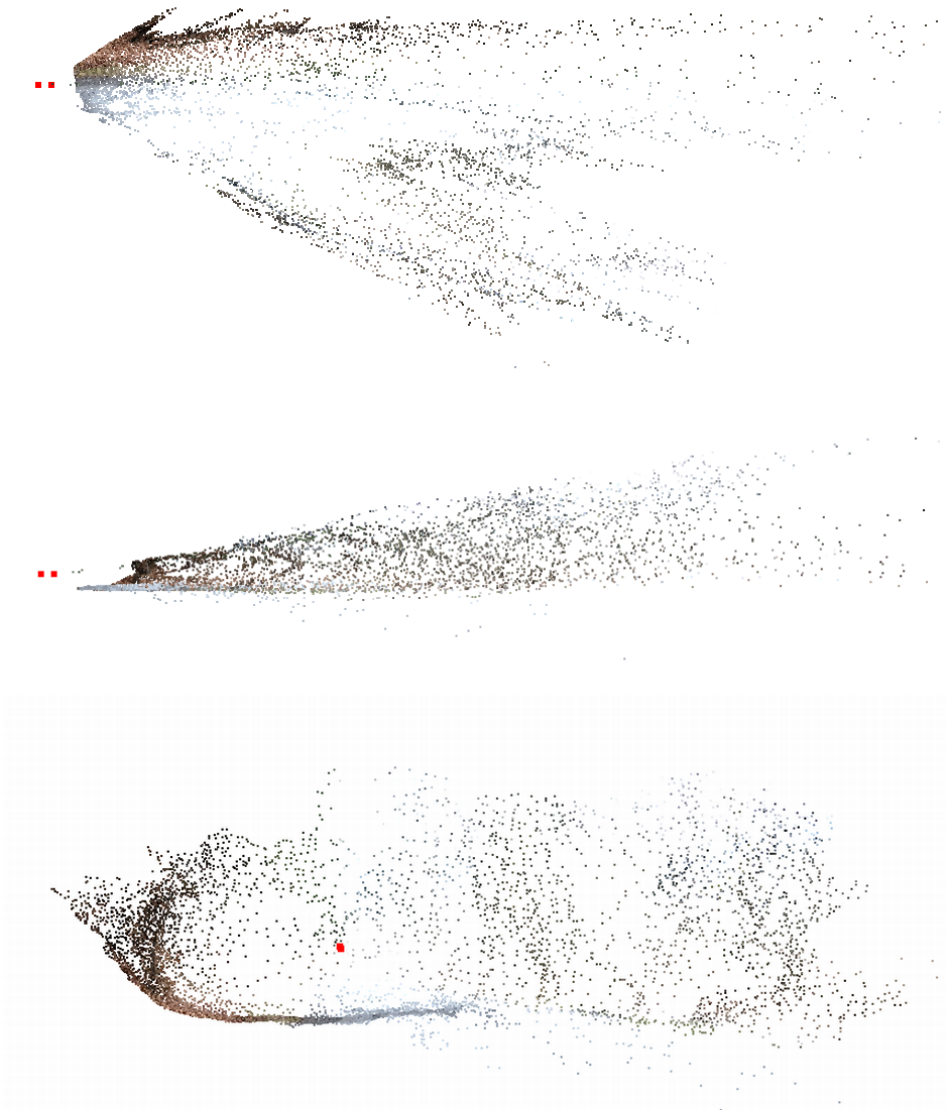
Vizualizace obou scén pomocí PCL vizualizátoru je pak vidět na obrázcích 25 a 26. Horní část obou obrázků znázorňuje scénu z horního pohledu, střední část scénu z bočního pravého pohledu a spodní část pohled zepředu. Velké červené body označují pozice kamer.

Na obrázku 25 je vidět rozptyl od kamery těch bodů, které se nacházely na středové čáře a krajnici, ale zejména bodů, které se nacházely na horním okraji vlevo od pozorovatele ležící budovy.

Stejná situace tzv. upadání okraje vozovky je vidět v dolní části obrázku 26 ve frontální vizualizaci scény.



Obrázek 25: 3D vizualizace městské scény



Obrázek 26: 3D vizualizace přílesní scény

6 Závěr

Cílem mé práce bylo navrhnout a implementovat jednotlivé části 3D rekonstrukce okolí vozidla a upozornit na případná úskalí.

Hlavním prvkem 3D rekonstrukce je výpočet dostatečně přesné fundamentální matice. Proto její výpočet probíhá z několika málo bodů, zpravidla v řádu desítek. Tyto body jsou vybírány tak, aby byly v obraze co nejlépe sledovatelné. Bohužel se však ve více než polovině případů stávalo, že výpočet fundamentální matice ovlivnily sice dobře sledovatelné body, ale takové, které neodpovídají bodu na povrchu reálného objektu. Takovými body jsou zpravidla body na rozhraní dvou od kamery různě vzdálených objektů. Fundamentální matice tak nebyla vypočtena správně a výsledná rekonstrukce tedy nemohla odpovídat skutečnosti. V takovém případě tedy musela proběhnout opakovaná rekonstrukce z následujících snímků sekvence.

V části detekce a sledování bodů se pak ukázalo být největším problémem sledování bodů náležících středové čáře a krajnicím silnice. Tyto body není možné napříč snímky dostatečně identifikovat a tak se většinou stává, že napříč snímky zůstávají v téže pozici vzhledem ke snímku. Neodpovídají tedy reálnému bodu ve snímané scéně a jejich rekonstrukce tedy neproběhne správně. Jedním z nejúčinnějších způsobů, jak rekonstrukci takových bodů zabránit je tyto body vůbec nedetekovat v takovýchto oblastech. K identifikaci takových oblastí by pak bylo žádoucí použít odpovídající algoritmus pro rozpoznání čar oddělujících jednotlivé jízdní pruhy.

Posledním hlavním úskalím při rekonstrukce se pak jeví mírně nepřesně vypočtená fundamentální matice, což se projevilo v oblasti epipólu v případě, kdy je pozice epipólu součástí snímků. Tyto oblasti jsou pak rekonstruovány s velkými nepřesnostmi. Možným řešením je pak opět vyřadit body náležící okolí epipólu z rekonstrukce.

Tomáš Worek

7 Reference

[1] Klein, Georg, & Murray, David, *Parallel Tracking and Mapping for Small AR Workspaces*, http://www.robots.ox.ac.uk/~gk/publications/KleinMurray2007ISMAR.pdf [online], 2007

[2] Klein, Georg, & Murray, David, *Parallel Tracking and Mapping on a Camera Phone*, http://www.robots.ox.ac.uk/~gk/publications/KleinMurray2009ISMAR.pdf [online], 2009

[3] Khvedchenya, Eugene, *Comparison of the OpenCV's feature detection algorithms*, http://computer-vision-talks.com/articles/2011-01-04-comparison-of-the-opencv-feature-detection-algorithms/ [online], 2011

[4] Kuntz, Noah, *OpenCV Tutorial 10 - CHapter 11*, http://dasl.mem.drexel.edu/~noahKuntz/openCVTut10.html [online], 2009

[5] Bouguet, Jean-Yves, *Pyramidal Implementation of the Lucas Kanade Feature Tracker*, http://robots.stanford.edu/cs223b04/algo_tracking.pdf [online]

[6] Hartley, Richard, *An Investigation of the Essential Matrix*, http://users.cecs.anu.edu.au/~hartley/Papers/Q/Q.pdf [online]

[7] Hartley, Richard, & Sturm, Peter, *Triangulation*, http://users.cecs.anu.edu.au/~hartley/Papers/triangulation/triangulation.pdf [online]

[8] Hartley, Richard, & Zisserman, Andrew, *Multiple View Geometry in Computer Vision, Second Edition, Chapter 8*, http://www.robots.ox.ac.uk/~vgg/hzbook/hzbook1/HZepipolar.pdf [online], 2004

[9] Olsson, Carl, *Computer Vision*, http://www.maths.lth.se/matematiklth/personal/calle/datorseende13/ [online], 2013

[10] Rosten, Edward, *FAST Corner Detection*, http://www.edwardrosten.com/work/fast.html [online]

[11] Shil, Roy, *Simple triangulation with OpenCV from Harley & Zisserman*, http://www.morethantechnical.com/2012/01/04/simple-triangulation-with-opencv-from-harley-zisserman-w-code/ [online], 2012